Genetic-based Decoder for Statistical Machine Translation

Douib Ameur
Langlois David
Smaïli Kamel

ameur.douib@inria.fr , david.langlois@loria.fr , kamel.smaili@loria.fr

Abstract

We propose a new algorithm for decoding on machine translation process. This approach is based on an evolutionary algorithm. We hope that this new method will constitute an alternative to Moses's decoder which is based on a beam search algorithm while the one we propose is based on the optimisation of a total solution. The results achieved are very encouraging in terms of measures and the proposed translations themselves are well built.

1 Introduction

In Statistical machine translation (SMT) [5] given a source sentence f, the system produces a translation e in the target language, which maximises the probability $P(e|f)$. SMT can be divided into three parts. The language model (LM), which estimates the fluency of the translation e in the target language. The translation model (TM), which estimates the quality of the translation (accuracy) e, given the source f. The last part of SMT system is the decoder, for which, the machine translation process can be considered as an optimisation issue, where it takes, in input, the source sentence f and uses the LM and the TM to produce the best possible translation e, from all possible translations.
Many algorithms are proposed to handle the issue of decoding. The first proposals used a word-based translation system [1, 8], where the alignment is between words. Nowadays, decoders use a phrase-based system [11], where the alignment is between phrases. MOSES is the most popular open source decoder used by the community [10]. It is based on a beam-search algorithm, where it builds incrementally a set of complete translations from partial translations and starting with the empty one. In the building process, new phrases are added for each partial translation hypothesis to produce new hypotheses. Consequently, a large number of hypotheses are produced. To

reduce this number, a pruning process is applied, where the n-best hypotheses are retained for the next step according to a score given by the LM and the TM. Finally, from the set of complete translation hypotheses, the one which has the highest score is chosen as the final translation e. This algorithm gives good results [9], but presents at least two drawbacks. The first one concerns the fact that it is impossible to challenge a previous decision of translation. That is why it is possible to miss a partial solution which could lead to the best final translation. The second one concerns the decision making. At each step, MOSES keeps some translation hypotheses and eliminates others, according to the scores of the partial translations. The final solution is made up on a series of decisions what we would like to challenge in our method.

Using a complete translation hypothesis from the beginning of the translation process can reduce the impact of these problems. With complete translation hypothesis, it is possible to visit each part of the research space and modify it if necessary.

In the literature, works have been proposed in order to achieve translations from a complete translation. In [12] the decoder starts with a complete translation and applies iteratively heuristics until it reaches the best solution. In each iteration, the neighbour translations are produced by applying some neighbour functions [12], which modify phrases and lead to new propositions of translation. This work gives good results but do not outperforms the state-of-the-art system.

In this paper, we propose a new decoder for SMT based on evolutionary algorithms, and more particularly those based on genetic principle [3, 6]. The advantage of the genetic algorithm is to use, not just one complete translation as in [12], but a population of complete translations. The combination of these translations, using crossover and mutation functions, produces more information allowing to take better decisions. The genetic algorithm is one of the most performant optimization algorithms [3], especially when the space search is huge.

In previous works, the genetic algorithm was proposed to handle some parts of the automatic translation. In [7] a genetic algorithm was used in a learning process to generate new translation examples, but for an example-based machine translation system and not for SMT. Another work [19] proposed an algorithm to generate a multi-word-based Translation Model, and to evaluate this model, a basic genetic algorithm was used as a translator. In this paper, we show the feasibility of using a genetic algorithm as a decoder for SMT. Also, we give a detailed adaptation of each component of the genetic algorithm.

In section 2, we define the Statistical Machine Translation problem. In section 3, we present a description of our genetic decoder. We present the corpus data and some comparative results in section 4. Finally, in section 5, we give the conclusion with the perspectives for future works.

2 Phrase-based statistical machine translation system

In phrase-based SMT the system takes a source sentence f=<f_1,f_2,...,$f_{|f|}$> in input, and the decoder produces the best possible translation e=<e_1,e_2,...,$e_{|e|}$> in the target language, where f_i (respectively e_i) is the i^{th} word in f (respectively e). The translation process segments f into phrases, translates each phrase into the target language, and reorder target phrases. The links between the source and target phrases define the alignment (a). The translation e must maximise the conditional probability $P(a, e|f)$ [18]. So, the problem is considered as an optimization issue:

$$\hat{e} = \underset{a,e}{argmax}[P(e) \times P(f|a, e)] \quad (1)$$

In equation (1) we distinguish the language model $P(e)$ and the phrase-based translation model $P(f|e)$. The decoder uses these two models as features, to evaluate the translations and finds the best one. Other features can be added to improve the evaluation.

3 Genetic algorithm for SMT

The basic idea of the genetic algorithm[3, 6] is to start with an initial population of solutions (chromosomes) and to produce iteratively new chromosomes using the crossover and the mutation functions. At the end of each iteration (generation) some chromosomes are selected from the population and kept for the next generation. This process ensures the evolution of the population towards a good solution (see Figure 1). An adequate representation of chromosomes and a good function to evaluate the chromosomes (fitness) are required to guarantee this evolution. Many parameters (population length, crossover and mutation rate, end process conditions, etc.) have to be fixed depending on the problem. In the next sections, we present how we adapt the genetic algorithm as a decoder for the SMT problem and we describe all the functions needed. We call our decoder GAMaT for Genetic Algorithm for Machine Translation.

3.1 Chromosome representation

A chromosome is composed of a serie of genes. A gene is an item of the translation hypothesis. As we use a phrase-based SMT, each gene will be associated to a phrase. This encoding makes easier the application of crossover and mutation functions at the phrase level. So, each chromosome c contains

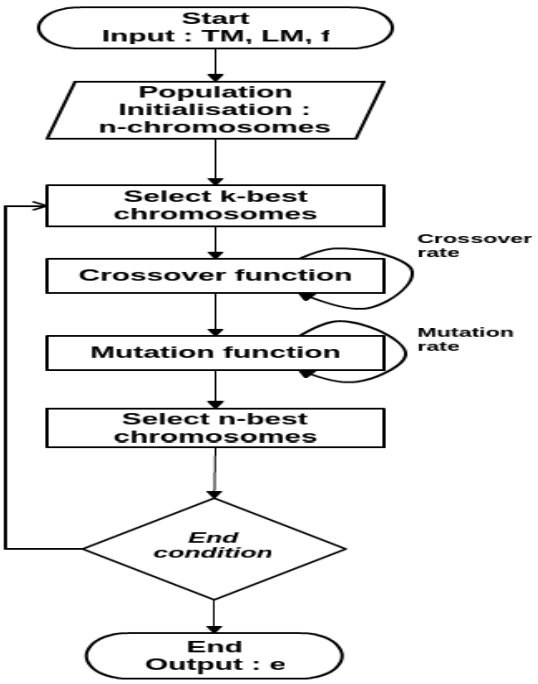

Figure 1 – Genetic Algorithm for SMT

the main following attributes:

- A complete translation hypothesis e.
- A phrase segmentation of f and e.
- An alignment a between the source and target phrases.

To simplify the notation for the next sections, we define a chromosome $c=<f, e, \overline{f}, \overline{e}, a>$, where $\overline{f}$ is the phrase segmentation of f, $\overline{e}$ is the phrase segmentation of e, and a the alignment between the source and target phrases. We denote f_i^j (respectively e_i^j) a phrase from f (respectively e) starting at position i and ending at position j. a_i is the position of the target phrase in $\overline{e}$ aligned with $\overline{f}_i$. When $a_i = i$ for every i, the alignment is monotone.

3.2 Initialisation functions

Five functions are used to produce the first population of chromosomes. All these functions, from the input sentence f, produce a phrase segmentation $\overline{f}$. After that, they take the best translation (target phrase) of each source phrase determined by the previous segmentation and add it to e. We use the Translation Table (TT) to select the target phrases. TT contains all information obtained from the training process, applied on a bilingual corpus

[13].
The first three functions produce the initial population by promoting longer phrases, this is useful since long phrases tend to cover more lexical and syntactic relationships between the different items of a solution. With these three functions, we produce three chromosomes. The two other functions use a random segmentation and have the objective to produce more chromosomes. For all the functions the alignment is monotone.
Before applying any function, we retrieve all possible phrases from f using TT, and save phrases' length information in the vector $PH =< ph_1, ..., ph_n >$ (Table 1), where ph_i is the length of the longest phrase in f starting at position i. We describe the initialization functions in the next sections.

Table 1 – Length phrases table

madame	la	président	,	la	prsidence	a	proclamé	le	résultat	du	vote	.
5	4	2	2	5	3	7	3	5	4	3	2	1

3.2.1 A left-to-right segmentation based on the maximum length of phrases

For this function, the source sentence f is segmented from left to right. The first phrase (segment) is the longest prefix of f that is present in TT. The remaining of f is processed using the same heuristic until f is entirely segmented. For each segment in f, we select from TT the English phrase with the highest probability. Then we concatenate these English phrases in order to obtain the initial translation. Figure 2 shows the result of this function applied to the example of Table 1.

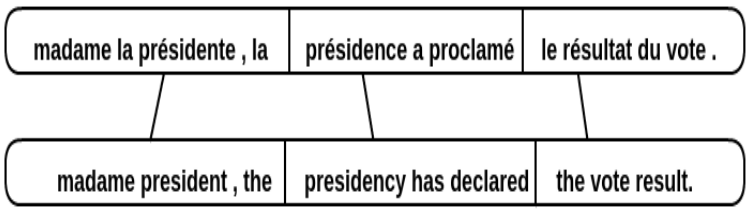

Figure 2 – Example of the left-to-right segmentation based on the maximum length of phrases

3.2.2 A right-to-left segmentation based on the maximum length of phrases

Contrary to the first function, here the segmentation is done from right to left. So, the first phrase is the longest suffix of f that is present in TT. The same process is applied until f is entirely segmented. The translation

e is generated in the same way as for the first function. Figure 3 shows the result of this initialization for the same example.

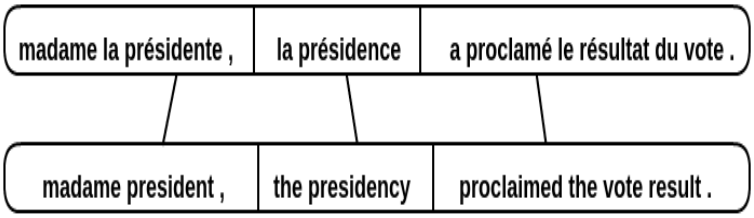

Figure 3 – Example of the right-to-left segmentation based on the maximum length of phrases

3.2.3 A global segmentation based on the maximum length of phrases

In this function non constraint is imposed to the direction of the segmentation. The main idea is to produce a segmentation with the minimum number of phrases. To produce this segmentation, we start by choosing the longest phrase $\overline{f}_i$ in f using the phrases' length information saved in PH. Then we apply recursively the same process for the left and right part of $\overline{f}_i$ until exhaustion of all items f. The Figure 4 shows the result of this segmentation using the PH.

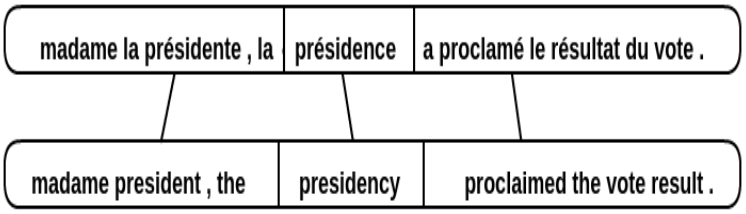

Figure 4 – Example of the global segmentation based on the maximum length of phrases

3.2.4 A left-to-right random segmentation

Like for the first function, the source sentence f is segmented from left to right. But, we select randomly from TT a phrase which is considered as a prefix of f. We apply the same process iteratively on the remaining of f. To produce the target sentence e we proceed in the same way as the previous functions. As this function takes random decisions, we use it to produce several chromosomes, one for each produced segmentation.

3.2.5 A right-to-left random segmentation

This second random segmentation function proceeds from right to left contrary to the previous one. Iteratively we choose a random suffix of f in the not yet segmented part of f as a new phrase, and which exists in TT. Like for the previous function, we use this function to generate several chromosomes.

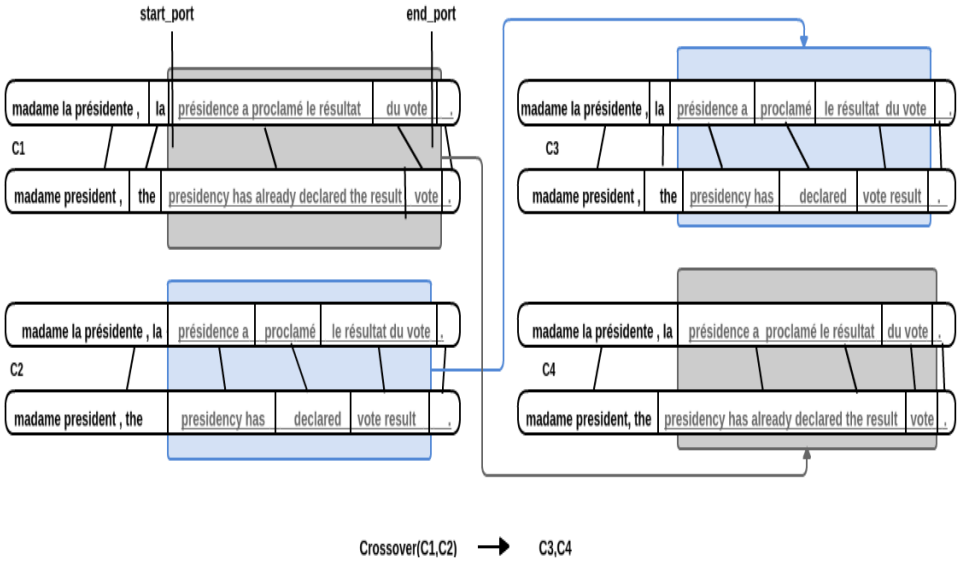

Figure 5 – Crossover function

3.3 Crossover function

A crossover operation consists in the following steps:
- Select randomly two chromosomes $c1$ and $c2$.
- Select randomly a subpart sp from f respecting some constraints: the first word of sp must be the first word of a segment from $c1.\overline{f}$ and $c2.\overline{f}$, and the last word of sp must be the last word of a segment from $c1.\overline{f}$ and $c2.\overline{f}$. In Figure 5, the chosen segment is *"présidence a proclamé le résultat du vote"*. This segment is a good candidate for crossover because *"présidence"* is the first word of $c1.\overline{f}_3$ and $c2.\overline{f}_2$, and "*vote*" is the last word of $c1.\overline{f}_4$ and $c2.\overline{f}_4$.
- Build $c3$ by taking sp from $c2$ (with its segmentation, its English counterpart, and the corresponding alignement). Complete $c3$ with the left and the

right parts from $c1$.
- Build $c4$ by taking sp from $c1$ (with its segmentation, its English counterpart, and the corresponding alignement). Complete $c4$ with the left and the right parts from $c2$.

3.4 Mutation functions

We use five mutation functions, which modify the aspect of an existing chromosome and allow consequently to produce a new one. These functions are presented below.

1. **Replace-Phrase**

   We choose randomly a source phrase, and replace its associated target phrase in e by another using TT. The new target phrase must have the highest probability of translation among all the possible translations excluding obviously the one we would like to mutate.

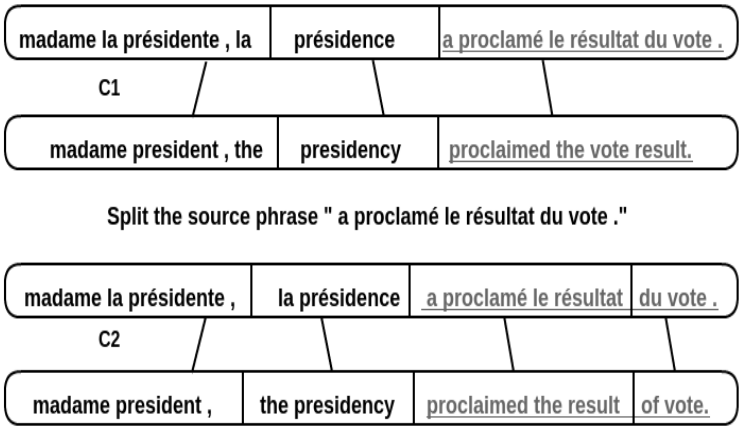

Figure 6 – Split mutation example

2. **Split-Phrase**

   Source phrase $\overline{f}_i$ is selectedand randomely and segmented into a left and a right part. These parts must be present in TT. Then, we look for the segmentation of the target phrase aligned with $\overline{f}_i$ such that the target left (respectively right) part can be aligned with the source left (respectively right) part of $\overline{f}_i$. If it is possible, we build a new chromosome with the updated segmentation and alignement. If not, the left and right parts of $\overline{f}_i$ are translated using the most probable

translations in TT. In Figure 6, the segmented source phrase is $c1.\overline{f}_3$. No translations are present in TT for *"a proclamé le résultat"* and *"du vote ."*, which exist separately in the aligned target phrase $c1.\overline{e}_{a_3}$. So, we choose new translation for each source phrase from TT.

3. **Merge-Two-Phrases**

   We choose randomly two adjacent phrases $\overline{f}_i$ and $\overline{f}_{i+1}$, where their aligned target phrases are also adjacent in the target sentence. The achieved phrase after merging $\overline{f}_i$ and $\overline{f}_{i+1}$ must exist in TT. If these conditions are met, we merge the selected source phrases into one phrase $\overline{f}_i$. We do the same process for the target phrases, if the result of this merging exists in TT as a translation of the new source phrase. Otherwise, we choose from the translation table a new target phrase $\overline{e}_{a_i}$ which has the highest translation probability. Finally, we update the segmentation and the alignment in the chromosome.

4. **Merge-Two-Phrases-and-Replace**

   This function has the same logic as Merge-Two-Phrases, except that we translate directly the new generated phrase using TT, without any verification.

5. **Swap-Two-Phrases**

   This last function selects randomly two adjacent source phrases $\overline{f}_i$ and $\overline{f}_{i+1}$. After that, we exchange the positions of $\overline{e}_{a_i}$ and $\overline{e}_{a_{i+1}}$ in the translation e. This mutation produces a non-monotone alignment.

3.5 Evaluation function : log-linear Model

Given a chromosome $c =< f, e, \overline{f}, \overline{e}, a >$, the log-linear approach is used to estimate the quality of the chromosome (translation). In this approach we calculate the logarithmic sum of a set of features functions $h_m(c)$, where m is the *id* of a feature. The result of this logarithmic sum represents the *score* of the chromosome.

$$Score(c) = \sum_m \lambda_m \times \log(h_m(\overline{e}, \overline{f}, a)) \tag{2}$$

Where λ_m is the weight of h_m. The value of each weight defines the influence of the corresponding feature in the final score. With this approach, we can add new features, to estimate better the translation. GAMaT uses the following features, which are the same for MOSES [10]:

– h_1: Language model probability

– h_2: Direct translation probability

– h_3: Inverse translation probability

– h_4: Direct lexical weighting probability

– h_5: Inverse Lexical weighting probability

– h_6: Phrase penalty

– h_7: Word penalty

– h_8: Reordering model

As we handle complete translations, the use of the word penalty in the same way as in Moses, the produced translations are shorter than the references. That is why, we define a new word penalty score, using a length translation model. In this model we define a length probability $P_l(e, f)$ for each pair of source and target sentence, which is estimated from the bilingual training corpus as follows:

$$P_l(e, f) = \frac{count(|f|, |e|)}{count(|e|)} \tag{3}$$

Where $count(|f|, |e|)$ is the number of times that source sentences of length $|f|$ has been translated by target translations of length $|e|$. $count(|e|)$ is the number of translations of length $|e|$ in the target corpus.

3.6 Selection process

The selection process is used in two cases, the first one, to define an *elite* set from which we will pick chromosomes as parents, to perform genetic manipulations. The *elite* set contains the k ($k < n$) best chromosomes of the population. The second selection is applied to select the n best chromosomes, which are kept for the next generation. The selection is based on the score of chromosomes.

4 Results and comparisons

4.1 Corpora

For our experiments, we use the 9^th task workshop on Statistical Machine Translation (2014) [4]. We take the French-English corpus to evaluate GAMaT and MOSES. The corpus contains 2,000,000 pairs of sentences. After a classical step of corpus preprocessing, we define the following three

sentences sets: *Train* set that contains 1,323,382 pairs of sentences for the training process. *Dev* set, that contains 165,422 pairs of sentences for the development process. Finally, *Test* set which contains 1,000 pairs of sentences to evaluate the decoders.
We use GIZA++ [13] to generate the translation model, and SRILM [17] for language model. MERT [2] is launched on the baseline system (MOSES) then we use the same weights in GAMaT since almost all the probability parameters are calculated by the tools associated to MOSES.

4.2 GAMaT parameters

In a genetic algorithm we have four important parameters: the number of chromosomes in the population (n), the number of chromosomes in the *elite* set, the crossover and mutation rates. We optimized n by varying its value and we found that 120 chromosomes in the population gives the best translation results. The *elite* set contains the best 75% chromosomes in the population. The crossover rate and the mutation rate fix the number of times which the crossover and mutation functions are applied in each generation. We optimized the crossover and mutation rates empirically. So, in the presented results the mutation and the crossover operations are applied at each iteration to 20% and 40% of the population respectively.

4.3 Comparative results

In this section, we present the performance of GAMaT, and we compare them to MOSES. To evaluate the quality of the translations achieved by the two decoders, we use BLEU [14] and TER [16] metrics. The first one calculates the number of n-gram which exists in the hypothesis and the reference translation at the same time. The second one, TER, calculates a cost of the modifications that we have to apply on hypothesis to obtain the reference translation.

Table 2 – GAMaT and MOSES performances using BLEU, TER

Decoder	BLEU	TER
MOSES	**29.32**	**53.02**
GAMaT-WPS	26.13	53.34
GAMaT-LtM	**27.15**	**53.08**
GAMaT+1-MOSES	27.22	52.81

In Table 2, we present the BLEU and TER scores for GAMaT and MOSES, where GAMaT-LtM represents GAMaT with the translation length

probabilty as feature, and GAMaT-WPS is GAMaT using the word penalty as feature. In the GAMaT+1-MOSES, we add the best translation of MOSES in the initialisation, as a chromosome. The underlying idea is to test if the 1-bets of MOSES can help GAMaT to produce better results.
The results show that there is a significant difference between GAMaT and MOSES, in terms of BLEU, however in terms of TER there, the two systems are equivalent. By studying the results, we mean the translations themselves achieved by both systems we found that GAMaT gives, in general, good translations compared to the source sentences. The main problem for GAMaT is the reordering. For some sentences, it has difficulties to apply the good permutations to find the best reordering in the target.
Adding the best translation of MOSES in the initialisation process does not improve significantly the result in terms of BLEU, however it outperforms MOSES in terms of TER which is very encouraging. It means that we have to improve the selection process in order to achieve more diversity in the population. So, improvement of the quality of the first population insures the improvement of the final translation.
Table 3 shows the number of translations that are better for a system than the other, in terms of TER. We can see that the two decoders give the same translation quality for 287 sentences from 1000. MOSES is better for 366 and GAMaT is better for 347 among them. From this result, we can deduce that each system manages to solve some translation problems better than the other.

Table 3 – Comparison between GAMaT and MOSES for each sentence, using TER

	Nbr-setences	%
Equal	287	28.70
MOSES better	366	36.60
GAMaT better	347	34.70

In the following, we will analyze certain functions of GAMaT in order to understand how to improve it. In Figure 7, we plot the evolution of the population in terms of BLEU. Each curve represents the evolution for one translation. The curves show clearly, that there is no stability in the evolution for the first iterations. This is normal because the population contains a large variety of chromosomes. But, after some iterations the population stabilizes and progresses until convergence. This evolution is a normal evolution for any genetic algorithm, and proves that the decoder manages to increase the quality of translation from an initial population.

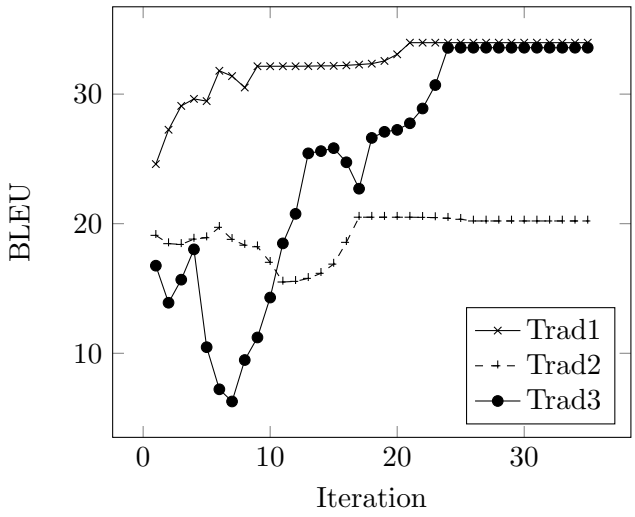

Figure 7 – Evolution of the population, using BLEU, for 3 different sentences

Finally, we analyse the influence of crossover and mutation functions in the research process. The values in Table 4 represent the number of times that these functions were used in the evolution of the best translations. We see that the crossover function is the most used function, with 67%, because this function is the main operation in any genetic algorithm, also because the fixed rate to apply the crossover function is 40%, compared with 20% for all the mutation functions. Also, we can see that the substitution of phrases is the most used mutation, through the *Replace-phrase* and *Merge-Two-Phrases-and-Replace* functions. The less used functions are *Split-Phrase* and *Swap-two-Phrases*, which is normal for the French-English pair, because the reordering can be covered by long segments. In order to fix the problem of reordering we tried to force GAMaT to use more the *Swap-two-phrases* function, but the results were not better.

Table 4 – Influence of crossover and mutation functions on final translations

Function	Nbr	%
Crossover	4491	67,92
M-Replace	858	12,98
M-Swap	345	5,22
M-Split	334	5,05
M-Merge	584	8,83
M-Merge+Replace	593	8,97

5 Conclusion and perspectives

In this paper, we presented GAMaT, a new decoder for the SMT. This decoder is based on a genetic algorithm which allows to use a set of complete translations. The obtained performance for the French-English pair are promising but not yet better than MOSES in terms of BLEU but they are equivalent in terms of TER. However, GAMaT succeeds to propose better translations in 34.7% of cases and gives identical results in 28,7% of cases. Analysing the produced translations, we noticed that GAMaT suffers from the mismanagement of the reordering process.
To outperform MOSES, we will mainly optimize the feature's weights independently from MOSES. In fact, we use exactly the same weights produced by MOSES, we have to set up them, for instance by an evolutionary algorithm. The confidence measures [15] also will be used in order to guide better the genetic operations.

References

[1] A. L. Berger, P. F. Brown, S. A. Della Pietra, V. J. Della Pietra, J. R. Gillett, J. D. Lafferty, R. L. Mercer, H. Printz, and L. Ures. The candide system for machine translation. *HLT '94 Proceedings of the workshop on Human Language Technology*, pages 157–162, 1994.

[2] N. Bertoldi, B. Haddow, and J-B. Fouet. Improved minimum error rate training in Moses. *The Prague Bulletin of Mathematical Linguistics*, pages 7–16, 2009.

[3] S. Binitha and S. Siva Sathya. A Survey of Bio inspired Optimization Algorithms. *International Journal of Soft Computing and Engineering (IJSCE)*, pages 2231–2307, 2012.

[4] O. Bojar, C. Buck, C. Federmann, B. Haddow, P. Koehn, J. Leveling, C. Monz, P. Pecina, M. Post, H. Saint-Amand, R. Soricut, L. Specia, and A. Tamchyna. Findings of the 2014 Workshop on Statistical Machine Translation. *ACL NINTH workshop on Statistical Machine Translation*, 2014.

[5] P. F. Brown, V. J. Della Pietra, S. A. Della Pietra, and R. L. Mercer. The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics*, pages 263–311, 1993.

[6] Matej Črepinšek, Shih-Hsi Liu, and Marjan Mernik. Exploration and exploitation in evolutionary algorithms: a survey. *ACM Computing Surveys (CSUR)*, 45(3):35, 2013.

[7] H. Echizen-ya, K. Araki, Y. Momouchi, and K. Tochinai. Machine translation method using inductive learning with genetic algorithms. *conference on Computational linguistics*, 2(16):1020–1023, 1996.

[8] U. Germann, M. Jahr, K. Knight, D. Marcu, and K. Yamada. Fast Decoding and Optimal Decoding for Machine Translation. *Artificial Intelligence 154*, pages 127–143, 2004.

[9] P. Koehn. A Beam Search Decoder for Phrase-Based Statistical Machine Translation Models. *Conference of the Association for Machine Translation in the Americas, AMTA*, pages 115–124, 2004.

[10] P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin, and E. Herbst. Moses: Open Source Toolkit for Statistical Machine Translation. *ACL '07 Proceedings of the 45th Annual Meeting of the ACL*, pages 177–180, 2007.

[11] P. Koehn, F. J. Och, and D. Marcu. Statistical phrase-based translation. *NAACL '03 Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 48–54, 2003.

[12] Philippe Langlais, Alexandre Patry, and Fabrizio Gotti. A greedy decoder for phrase-based statistical machine translation. *Proc. of TMI*, 2007.

[13] F. J. Och and H. Ney. A systematic comparison of various statistical alignment models. *Computational Linguistics*, pages 19–51, 2003.

[14] K. Papineni, S. Roukos, T. Ward, and W-J. Zhu. BLEU: a method for automatic evaluation of machine translation. *ACL '02 Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 311–318, 202.

[15] S. Raybaud, D. Langlois, and K. Smaili. 'this sentence is wrong.' Detecting errors in machine-translated sentences. *Machine Translation*, pages 1–34, 2011.

[16] M. Snover, B. Dorr, R. Schwartz, and et. A study of translation edit rate with targeted human annotation. *In : Proceedings of the Association for Machine Translation in the Americas*, pages 223–231, 2006.

[17] A. Stolcke, J. Zheng, W. Wang, and V. Abrash. SRILM at Sixteen: Update and Outlook. *Proceedings of IEEE Automatic Speech Recognition and Understanding Workshop*, page 5, 2011.

[18] R. Zens, F. J. Och, and H. Ney. Phrase-Based Statistical Machine Translation. *Human language Technology and Pattern Recognition*, pages 18–32, 2002.

[19] A. Zogheib. Genetic Algorithm-based Multi-Word Automatic Language Translation. *Recent Advances in Intelligent Information Systems*, pages 751–760, 2011.